10. Richard

Program Name: Richard.java **Input File: richard.dat**

Richard has some really bad news! His personal server was just hacked! Richard isn't sure exactly how the server was hacked, but he thinks his old password of "password123" might be the issue. In an attempt to better safeguard his server, Richard has the idea to use a password generator to create a much stronger, and harder to crack password. The problem is, Richard no longer trusts anyone or any application created by anyone other than himself or someone he knows personally!

With this said, Richard wants to write a program that generates a password and he has the idea to take a given string of characters made up of letters (uppercase and lowercase), digits, and special characters and determine the ith permutation of that string. For example, suppose Richard chooses the string "`AbC!`". If Richard were to generate all possible permutations of this string and sort them by their ASCII values he would have: `"!ACb", "!AbC", "!CAb", "!CbA", "!bAC", "!bCA", "A!Cb", "A!bC", "AC!b", "ACb!", "Ab!C". "AbC!", "C!Ab", "C!bA", "CA!b", "CAb!", "Cb!A", "CbA!", " b!AC", "b!CA", "bA!C", "bAC!", "bC!A", "bCA!".` So, if Richard chose his password to be the 16th permutation of the string "`AbC!`", his password would be: `"CAb!"`

Can you assist Richard in writing a program to generate new passwords for his server?

Input: The input will consist of an integer *T*, the number of test cases. *T* will be in the range of [1,20]. For each test case, input will consist of two lines. Line 1 will be a number `i`, in range [1, 2147483647]. This is the ith permutation number Richard has selected. Line 2 will consist of a string of characters (upper and lower case), digits, and printable special characters. The length of the string will be guaranteed to be at least one character long and no more than 50 characters long. The string is also guaranteed to have no two characters with the same ASCII value, and the ith permutation is guaranteed to exist.

Output: For each test case, you are to output `"Password #X: GENERATED_PASSWORD"` where `X` is the test case number and `GENERATED_PASSWORD` is the ith permutation of the given string of characters.

Sample input:
```
6
16
AbC!
2
abcde
598
AaBbcC!
104567
t%&avcdef3
1
zAaBb
123456
8rTv3@AM!
```

Sample output:
```
Password #1: CAb!
Password #2: abced
Password #3: !bcaBCA
Password #4: %aet3cvf&d
Password #5: ABabz
Password #6: @!MArvT83
```